

Online Buy-at-Bulk Network Design

Deeparnab Chakrabarty* Alina Ene† Ravishankar Krishnaswamy‡
Debmalaya Panigrahi§

Abstract

We present the first non-trivial online algorithms for the non-uniform, multicommodity buy-at-bulk (MC-BB) network design problem. Our competitive ratios qualitatively match the best known approximation factors for the corresponding offline problems. In particular, we show

- A polynomial time online algorithm with a poly-logarithmic competitive ratio for the MC-BB problem in undirected edge-weighted graphs.
- A quasi-polynomial time online algorithm with a poly-logarithmic competitive ratio for the MC-BB problem in undirected node-weighted graphs.
- For any fixed $\epsilon > 0$, a polynomial time online algorithm with a competitive ratio of $\tilde{O}(k^{\frac{1}{2}+\epsilon})$ (where k is the number of demands, and $\tilde{O}(\cdot)$ hides polylog factors) for MC-BB in directed graphs.
- Algorithms with matching competitive ratios for the prize-collecting variants of all the above problems.

Prior to our work, a logarithmic competitive ratio was known for undirected, edge-weighted graphs only for the special case of *uniform* costs (Awerbuch and Azar, FOCS 1997), and a polylogarithmic competitive ratio was known for the edge-weighted *single-sink* problem (Meyerson, SPAA 2004). To the best of our knowledge, no previous online algorithm was known, even for uniform costs, in the node-weighted and directed settings.

Our main engine for the results above is an *online reduction theorem* of MC-BB problems to their single-sink (SS-BB) counterparts. We use the concept of *junction-tree solutions* (Chekuri *et al.*, FOCS 2006) that play an important role in solving the *offline* versions of the problem via a greedy subroutine – an inherently offline procedure. Our main technical contribution is in designing an online algorithm using only the *existence* of good junction-trees to reduce an MC-BB instance to multiple SS-BB subinstances. Along the way, we also give the first non-trivial online node-weighted/directed single-sink buy-at-bulk algorithms. In addition to the new results, our generic reduction also yields new proofs of recent results for the online node-weighted Steiner forest and online group Steiner forest problems.

*Microsoft Research, 9 Lavelle Road, Bangalore, India. Email: dechakr@microsoft.com.

†Department of Computer Science and DIMAP, University of Warwick, Coventry, UK. Email: A.Ene@warwick.ac.uk.

‡Microsoft Research, 9 Lavelle Road, Bangalore, India. Email: rakri@microsoft.com.

§Department of Computer Science, Duke University, Durham, NC, USA. Email: debmalaya@cs.duke.edu.

1 Introduction

In a typical network design problem, one has to find a minimum cost (sub) network satisfying various connectivity and routing requirements. These are fundamental problems in combinatorial optimization, operations research, and computer science. To model economies of scale in network design, Salman *et al.* [32] proposed the *buy-at-bulk* framework, which has been studied extensively over the last two decades (e.g., [6, 20, 33, 22, 29, 28, 15, 14]). In this framework, each network element is associated with a sub-additive function representing the cost for a given utilization. Given a set of connectivity demands comprising k source-sink pairs, the goal is to route integral flows from the sources to the corresponding sinks concurrently to minimize the total cost of the routing.

An important application of the problem is capacity planning in telecommunication networks or in the Internet. As observed by Awerbuch and Azar [6], this application is inherently “online” in that terminal-pairs arrive over time and need to be served without knowledge of future pairs. The authors of [6] give a logarithmic-competitive online algorithm for the *uniform* case where every edge is associated with the same cost function. However, uniformity is not always a feasible assumption, especially in heterogeneous, dynamic networks like the Internet. Indeed recent research (e.g., [29, 28, 15]) has focused on the non-uniform setting with a different sub-additive function for every network element. In this non-uniform setting, Meyerson [28] gives a polylogarithmic-competitive algorithm for the special case when all terminal-pairs share the same sink. To the best of our knowledge, no non-trivial online algorithm is known for the general *multicommodity* setting, which is the focus of our paper.

We consider, in increasing order of generality, *undirected edge-weighted* graphs, *undirected node-weighted* graphs, and *directed edge-weighted* graphs.¹ It is also convenient to classify the problems that we study into the *single-sink* version where all the terminal-pairs share a common sink, and the general *multicommodity* version where the sinks in the terminal-pairs may be distinct. For notational convenience, we use the following shorthand forms for our problems: X-Y-BB where X = SS or MC (single-sink and multicommodity, respectively) and Y = E or N or D (undirected edge-weighted, undirected node-weighted, and the general directed case, respectively).

1.1 Our Contributions

We obtain the following new results (unless otherwise noted, our algorithms run in polynomial time):

- A poly-logarithmic competitive online algorithm for the MC-E-BB problem.
- A poly-logarithmic competitive online algorithm for MC-N-BB and SS-N-BB that runs in quasi-polynomial time.
- An $\tilde{O}(k^{\frac{1}{2}+\varepsilon})$ -competitive online algorithm for MC-D-BB for any constant $\varepsilon > 0$ with running time $n^{O(1/\varepsilon)}$, where $\tilde{O}(\cdot)$ hides polylogarithmic factors. For SS-D-BB, the ratio improves to $\tilde{O}(k^\varepsilon)$, translating to a polylogarithmic competitive ratio in quasi-polynomial time.
- Online algorithms for prize-collecting versions of all the above problems with the same competitive ratio.

Up to exponents in the logarithm, our online algorithms match the best known *offline* approximation algorithms (Chekuri *et al.* [15] for MC-E/N-BB and Antonakopoulos [5] for MC-D-BB); for MC-N-BB,

¹In undirected graphs, node costs can simulate edge costs; in directed graphs they are equivalent.

however, a *polynomial time*, polylogarithmic approximation is known [14], whereas our algorithm runs in quasi-polynomial time. Furthermore, a logarithmic lower bound, even for SS-E-BB, follows from the lower bound for the online Steiner tree problem [26], and a polylogarithmic lower bound for online SS-N-BB follows from a matching one for set cover [3].

From a technical perspective, we derive all the multicommodity results using a generic online reduction theorem that reduces a multicommodity instance to several single-sink instances, for which we either use existing online algorithms or give new online algorithms. Informally, one can view this as the “online analog” of the junction-tree approach pioneered by Chekuri *et al.* [15] for offline multicommodity network design. We discuss this approach in the next subsection.

1.2 An Online Reduction to Single Sink Instances

Multicommodity network design problems, both online and offline, are typically more challenging than their single-sink counterparts, and have historically² required new ideas every time depending on the specific problem at hand. The situation is no different for buy-at-bulk, both for uniform and non-uniform costs.

In the offline buy-at-bulk setting, this shortcoming is addressed by Chekuri *et al.* [15] (expanded to other problems by [14, 13, 5]), who introduce a generic combinatorial framework for mapping a single instance of a multicommodity problem to multiple instances of the corresponding single-sink problem. At the heart of this scheme is the following observation that holds for many multicommodity problems such as (edge/node) Steiner forest, directed Steiner network, buy-at-bulk, and set connectivity: there exists a near-optimal³ *junction-tree* solution for the multicommodity problem that decomposes into solutions to multiple single-sink problems where each single-sink problem connects some subset of the original terminal-pairs to a particular root.

The problem now reduces to finding good junction-trees to cover all the terminal-pairs. The offline techniques [15, 13, 5] tackle this using a greedy algorithm for finding the single-sink solutions; more precisely, in each step they find the best density (cost per terminal-pair) solution that routes a subset of terminal-pairs via a single sink. A set cover style analysis then bounds the loss for repeating this procedure until all terminal-pairs are covered. However, as the reader may have already noticed, the greedy optimization approach is inherently offline, as finding the best-density solution requires knowledge of all terminal-pairs upfront. Our main technical contribution in this work is *an online version of the junction-tree framework*. Indeed, we show how to reduce any multicommodity buy-at-bulk instance to a collection of single-sink instances online.

(Informal Theorem) If the junction-tree approximation factor of the MC-BB problem is α , the integrality gap of a natural LP relaxation of the SS-BB problem is β , and there is a γ -competitive online algorithm for the SS-BB problem, then there is an $O(\alpha\beta\gamma \cdot \text{polylog}(n))$ -competitive algorithm for the MC-BB problem.

To prove the above theorem, we first write a composite-LP, which has (a) an *outer-LP* comprising assignment variables that fractionally assign terminal-pairs to roots, and (b) many *inner-LPs* which correspond to the natural LP relaxations for the SS-BB problem for each root and the terminal-pairs fractionally assigned to it by the outer-LP. We then apply the framework of online primal-dual algorithms (see [10] for instance) to solve the composite-LP online. However, there are two main challenges we need to surmount.

- First, the existing framework has been mostly applied to purely covering/packing LPs⁴ and our inner-

²For instance, compare [29] and [12] for the SS-E-BB and MC-E-BB problem, and compare Naor *et al.* [31] and Hajiaghayi *et al.* [24] for the online node-weighted Steiner tree and Steiner forest problem.

³We call the quality of such a solution the junction-tree approximation factor; e.g., it is $O(\log n)$ for MC-E-BB and MC-N-BB [15]

⁴Our current understanding of mixed packing-covering is limited [7] and does not capture the problem we want to solve.

LPs have both kinds of constraints, and moreover, there is an outer-LP encapsulating them. We show nonetheless that it can be extended to solving our LP fractionally up to polylogarithmic factors. Indeed, we use the specific flow-structure of the inner-LP, and each step of our algorithm solves many (auxiliary) min-cost max-flow problems.

- The second difficulty is in *rounding* this fractional solution online. This is a hard problem, and currently we do not know how to do so even for basic network design problems such as the Steiner tree problem. To circumvent this, we show that it suffices to only *partially round* the LP. More precisely, we round the LP solution so that only the outer-LP (assignment variables) become integral, and the inner-LPs remain fractional. This gives us an *integral assignment* of the terminal-pairs to different single-sink instances, with bounded total fractional cost. Now, from the bounded integrality gap of the inner-LPs, we know that *there exist good single-sink solutions* for our assignment of terminal-pairs to roots, even though we cannot find them online⁵. Using this knowledge, our final step is to run online single-sink algorithms for each root, and send the terminal-pairs to the root as determined by the outer-LP assignment. Figure 1 summarizes our overall approach.

initialize multiple online algorithms for the single-sink problem (one for each vertex as root).
when (s_i, t_i) arrives

update the fractional solution of the composite LP to satisfy the new request

round the composite LP to get an integral solution to the outer LP, which gives us an assignment of (s_i, t_i) to some root r

send both s_i and t_i to the instance of the single-sink online algorithm with root r

Figure 1: **Online Framework for Multicommodity Network Design Problems**

The results mentioned in Section 1.1 follow by bounding α, β, γ for the corresponding problems. For MC-E-BB, all of these are known to be $O(\text{polylog}(n))$ ([15, 29, 28] respectively). For MC-N-BB, it is known both α, β are bounded by $O(\text{polylog}(n))$ [14], and we bound γ in Section 5 by giving the first online algorithms for SS-N-BB. For MC-D-BB, we need some additional work. In this case we cannot directly bound β , since the integrality gap of the natural LP relaxation is *not known* to be bounded. Nevertheless, in Section 4, we show that it suffices to work only with more structured instances for which we can bound the integrality gap.

Finally, we illustrate the generality of our reduction theorem by noting that, when combined with existing bounds on α, β , and γ , it immediately implies (up to polylogarithmic factors) some recent results in online network design, such as online node-weighted Steiner forest [24], and online edge-weighted group Steiner forest [31] – two problems for which specialized techniques were needed, even though their single-sink counterparts were known earlier.

⁵The difficulty comes from the fact that the online solution we maintain must be monotonic, i.e., the decisions are irrevocable.

1.3 Related Work

Buy-at-bulk network design problems have received considerable attention over the last two decades, both in the offline and online settings. For the uniform cost model, Awerbuch and Azar [6] give an $O(\log n)$ -approximation for MC-E-BB, while $O(1)$ -approximations are known [20, 33, 22] for SS-E-BB. We also note that $O(1)$ -approximations have been obtained in special cases for the multicommodity problem, such as in the *rent-or-buy* setting [21]. Meyerson *et al.* [29] give an $O(\log k)$ approximation for the general SS-E-BB, and the first non-trivial algorithm for MC-E-BB is an $\exp(O(\sqrt{\log n \log \log n}))$ -approximation due to Charikar and Karagiozova [12]. This was improved to a poly-logarithmic factor by Chekuri *et al.* [15] who also solve MC-N-BB [14] with similar guarantees. For directed graphs, our knowledge is much sparser. Even for special cases like directed Steiner tree and forest, the best polytime approximation factors known are $O(k^\varepsilon)$ [34, 11] and $\min(O(\sqrt{k}), n^{2/3})$ [13, 18, 8] respectively, and these ideas were extended to MC-D-BB by Antonakopoulos [5]. On the hardness side, Andrews [4] shows that even the MC-E-BB problem is $\Omega(\log^{1/2-\varepsilon} n)$ -hard, while MC-D-BB (in fact directed Steiner forest) is known to be label-cover hard [17].

The online Steiner tree problem (a special case of online SS-E-BB) was first studied by Imase and Waxman [26] who give an $O(\log k)$ -competitive algorithm. Berman and Coulston [9] give an $O(\log k)$ -competitive algorithm for online Steiner forest, and both these results are tight, i.e., there is an $\Omega(\log k)$ lower bound. As mentioned earlier, Awerbuch and Azar’s algorithm [6] can be seen as an $O(\log n)$ -competitive online algorithm for the *uniform-cost* MC-E-BB. For non-uniform buy-at-bulk, the only online algorithm that we are aware of is Meyerson’s [28] polylog-competitive algorithm for the single-sink problem. For online node-weighted network design, developments are much more recent. A polylogarithmic approximation for the node-weighted Steiner tree problem was first given by Naor *et al.* [31] and later extended to the online node-weighted Steiner forest problem [24] and prize-collecting versions [23]. These algorithms, like ours in this paper, utilize the online adaptation of the primal-dual and LP rounding schemas pioneered by the work of Alon *et al.* [3] for the online set cover problem (see also [2] for its adaptation to network design problems). We also note that in the node-weighted setting, the online lower bound can be strengthened to $\Omega(\log n \log k)$ using online set cover lower bounds [3, 27].

2 Preliminaries and Results

We now formally state the problem, set up notation that we use throughout the paper, and state our main theorems.

2.1 Our Problems

Buy-at-bulk Network Design. In the most general setting of the MC-D-BB problem, an instance $\mathcal{I} = (G, \mathcal{X})$ consists of a directed graph $G = (V, E)$ and a collection \mathcal{X} of *terminal-pairs* $(s_i, t_i) \in V \times V$; each such s_i and t_i is called a terminal. Each (s_i, t_i) pair also has a positive integer demand d_i , which we assume to be 1 for clarity in presentation.⁶ Additionally, each edge $e \in E$ is associated with a *monotone, sub-additive*⁷ cost function $f_e : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$. A feasible solution to the problem is a collection of paths $\{P_1, \dots, P_k\}$ where P_i is a directed path from s_i to t_i carrying load d_i . Given a solution $\{P_1, \dots, P_k\}$, we

⁶We can handle non-uniform demands by incurring an additional $O(\log D)$ factor in the competitive ratio and the running time (where D is the maximum demand) by having $O(\log D)$ “unit-demand” instances, where the i^{th} instance deals with demands between 2^{i-1} and 2^i .

⁷That is, $f_e(x) \geq f_e(y)$ whenever $x \geq y$, and $f_e(x+y) \leq f_e(x) + f_e(y)$

let $\text{load}(e) = \sum_{i: e \in P_i} d_i$ denote the total load on edge e . The goal is to find a feasible solution minimizing the objective $\text{Obj}_{\text{BB}} := \sum_{e \in E} f_e(\text{load}(e))$. In the **online** problem, the offline input consists of the graph G and the cost functions f_e . The pairs (s_i, t_i) arrive online in an unknown, possibly adversarial, order. When a pair (s_i, t_i) arrives, the algorithm must select the path P_i that connects them, and this decision is irrevocable.

Reduction to the Two-metric Problem. Following previous work, throughout this paper we consider an equivalent problem (up to constant factors) known as **two-metric network design**. In this problem, instead of functions $f_e(\cdot)$ on the edges, we are given two parameters c_e and ℓ_e on each edge. One can think of c_e as a fixed buying cost, or just *cost*, of edge e , and ℓ_e as a per-unit flow cost, or *length*, of edge e . The feasible solution space is the same as for the buy-at-bulk problem, and the goal is to minimize the objective $\text{Obj}_{2\text{M}} := \sum_{e \in \bigcup_i P_i} c_e + \sum_i \sum_{e \in P_i} \ell_e$. The following lemma is well known (see e.g., [15]).

Lemma 1. *Given an instance of the buy-at-bulk problem, for any $\varepsilon > 0$ one can find an instance of the two-metric network design problem such that, for any feasible solution, $\text{Obj}_{2\text{M}} \leq \text{Obj}_{\text{BB}} \leq (2 + \varepsilon)\text{Obj}_{2\text{M}}$.*

Remark. *In light of the above lemma, henceforth we abuse notation and let the buy-at-bulk problem mean the two-metric network design problem.*

2.2 Our Tools

Junction-tree solutions. Given an instance $\mathcal{I} = (G, \mathcal{X})$ of the buy-at-bulk problem, we consider *junction-tree*⁸ solutions, a specific kind of solution to the problem introduced by [15]. In such solutions, the collection of pairs are partitioned into groups and each group is indexed by a *root vertex* $r \in V$. For all terminal pairs (s_i, t_i) in a group indexed by r , the path P_i from s_i to t_i contains the root vertex r (see Figure 2).

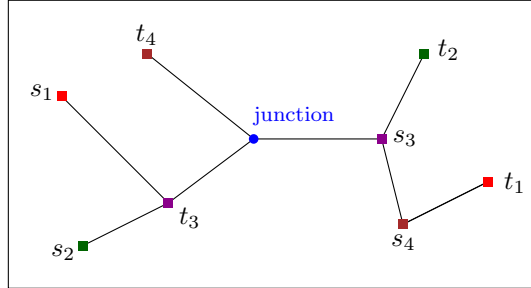


Figure 2: A group of terminal pairs routed via a junction-vertex in an undirected graph.

Formally, consider an instance $\mathcal{I} = (G, \mathcal{X})$ of the buy-at-bulk problem, and let Opt denote the objective value of the optimum solution. Given a partition $\Pi := (\pi_{r_1}, \dots, \pi_{r_q})$ of terminal pairs indexed by q different root vertices, a junction-tree solution is one that uses *single-sink* solutions to connect the original terminal-pairs. Indeed, for each part π_r indexed by root r , consider the optimal solutions to the single-sink problem on graph G with demands $\{(s_i, r) : (s_i, t_i) \in \pi_r\}$ and the single-source problem⁹ with pairs $\{(r, t_i) : (s_i, t_i) \in \pi_r\}$. Let $\text{Opt}_r(\pi_r)$ denote the sum of the objectives of the optimal solutions to the single-sink and single-source problems, and let $\text{Opt}(\Pi) := \sum_{r \in V} \text{Opt}_r(\pi_r)$. Let Opt_{junc} denote the minimum $\text{Opt}(\Pi)$

⁸The word tree is misleading since the final solution need not be a tree in directed graphs. Nevertheless, we continue using this term for historical reasons. Junction trees were originally proposed for undirected graphs, where the solution is indeed a tree.

⁹The single-source problem in a directed graph is identical to the single-sink problem with all the edges reversed in direction. For undirected graphs, both problems are on the same graph.

over all partitions. We call this solution the *optimum junction-tree* solution for this instance.¹⁰ Clearly, $\text{Opt}_{\text{junc}} \geq \text{Opt}$. The **junction-tree approximation factor** of \mathcal{I} is defined to be the ratio $\text{Opt}_{\text{junc}}/\text{Opt}$.

LP Relaxation. We now describe a natural flow-based LP relaxation for the *single-sink* buy-at-bulk problem for an instance $\mathcal{I} = (G, \mathcal{T})$ where \mathcal{T} is a set of terminals that need to be connected to the root r .

$$\begin{aligned}
& \text{minimize} && \sum_{e \in E} c_e x_e + \sum_i \sum_{e \in E} \ell_e f_i(e) && \text{(SS-BaB LP)} \\
& \text{s.t} && \{f_i(e) : e \in E(G)\} \text{ defines a flow from } s_i \text{ to } r \text{ of value 1} && \forall s_i \in \mathcal{T} \\
& && f_i(e) \leq x_e && \forall e \in E \\
& && x_e \geq 0, f_i(e) \geq 0 && \forall e \in E
\end{aligned}$$

Recall that the *integrality gap* of (SS-BaB LP) on the instance $\mathcal{I} = (G, \mathcal{T})$ is defined to be the ratio of Opt to the optimal value of the LP (SS-BaB LP). Also, we define the integrality gap for the graph G to be the worst case integrality gap (over all requests \mathcal{T} on graph G) of the corresponding instance $\mathcal{I} = (G, \mathcal{T})$.

2.3 Our Results

Main Technical Theorem and its Applications. Now we are ready to state our main theorem; the proof is in Section 3. We say that an online algorithm is γ -competitive for a graph G if, for any sequence of requests \mathcal{X} , the online algorithm for buy-at-bulk returns a solution within a γ -factor of $\text{Opt}(\mathcal{I})$, where $\mathcal{I} = (G, \mathcal{X})$.

Theorem 2 (Reduction to Single-Sink Online Algorithms). *Fix an instance $\mathcal{I} = (G, \mathcal{X})$ of the MC-BB problem. Suppose the following three conditions hold.*

- (i) *The junction-tree approximation factor of \mathcal{I} is at most α .*
- (ii) *The integrality gap of (SS-BaB LP) on any single-sink instance on graph G is at most β .*
- (iii) *There is a γ -competitive online SS-BB algorithm for any instance on graph G that runs in time T .*

Then there is an online algorithm for \mathcal{I} running in time $\text{poly}(n, T)$ whose competitive ratio is $O(\alpha\beta\gamma \cdot \text{polylog}(n))$.

Using this theorem, we can immediately obtain the following new results mentioned in the introduction.

Theorem 3 (Undirected Edge-weighted Buy-at-Bulk). *There is a $\text{polylog}(n)$ -competitive, polynomial time randomized online algorithm for the MC-E-BB problem.*

Proof: The theorem follows directly by combining Theorem 2 with the following results from previous work. Chekuri *et al.* [15] prove that the junction-tree approximation factor for the undirected edge-weighted buy-at-bulk problem is $O(\log k)$. Chekuri *et al.* [16] prove that the integrality gap of (SS-BaB LP) in undirected edge-weighted graphs is $O(\log k)$. Meyerson [28] gives a randomized polynomial time online algorithm for the single-sink buy-at-bulk problem with competitive ratio $O(\log^4 n)$. ■

Theorem 4 (Undirected Node-weighted Buy-at-Bulk). *For any constant $\varepsilon > 0$, there is an $O(k^\varepsilon \text{polylog}(n))$ -competitive, randomized online algorithm for MC-N-BB with running time $n^{O(1/\varepsilon)}$. As a corollary, this yields a $\text{polylog}(n)$ -competitive, quasi-polynomial time algorithm for this problem.*

¹⁰Note that copies of the same edge appearing in multiple single-sink solutions are treated as distinct edges in the junction-tree solution. Hence, decomposing the optimal multicommodity solution into its constituent paths does not yield $\text{Opt}_{\text{junc}} = \text{Opt}$.

Theorem 5 (Directed Buy-at-Bulk). *For any constant $\varepsilon > 0$, there is an $O(k^{1/2+\varepsilon} \text{polylog}(n))$ -competitive, polynomial time online algorithm for the MC-D-BB.*

We again use Theorem 2 to prove the above theorems. However, unlike for MC-E-BB, we are not aware of any online algorithms for the SS-N-BB and SS-D-BB problems. We therefore first give online algorithms for these problems, and then use Theorem 2; the details appear in Sections 5 and 4.

Finally, we can almost directly use Theorem 2 to also obtain matching results for prize-collecting versions of the above problems. Recall that in a prize-collecting problem, every terminal-pair also comes with a penalty q_i , and the algorithm can opt to not satisfy the request by incurring this value in the objective. We give the extension of our results to the corresponding prize-collecting problems in Section 6.

Theorem 6. *For each of the above problems, there is an online algorithm with matching running time and competitive ratio for the corresponding prize-collecting version.*

In addition to the new results mentioned above, we can also use Theorem 2 to give alternative proofs (with slightly worse polylog factors) of some recent results in online network design. By combining Theorem 2 with the $\text{polylog}(n)$ -competitive algorithm for online group Steiner Tree due to Alon *et al.* [2], we obtain a $\text{polylog}(n)$ -competitive online algorithm for the group Steiner forest problem – a result shown earlier by Naor *et al.* [31]. Similarly, by combining Theorem 2 with the $\text{polylog}(n)$ -competitive online algorithm for the node-weighted Steiner tree problem due to Naor *et al.* [31], we obtain a $\text{polylog}(n)$ -competitive online algorithm for the node-weighted Steiner forest problem – a result shown earlier by Hajiaghayi *et al.* [24].

Height Reduction Theorem. One of the technical tools that we use repeatedly in this paper is the following result, which builds on the work of Helvig *et al.* [25]. We give the proof in Appendix A.

Theorem 7. *Given a directed graph $G = (V, E)$ with edge costs c_e and lengths ℓ_e , for all $h > 0$, we can efficiently find an upward directed, layered graph G_h^{up} on $(h + 1)$ levels and edges (with new costs and lengths) only between successive levels going from bottom (level h) to top (level 0), such that each layer has n vertices corresponding to the vertices of G , and, for any set of terminals X and any root vertex r ,*

- (i) *the optimal objective value of the single-sink buy-at-bulk problem to connect X (at level h) with r (at level 0) on the graph G_h^{up} is at most $O(hk^{1/h})\phi$, where ϕ is the objective value of an optimal solution of the same instance on the original graph G ;*
- (ii) *given a integral (resp. fractional solution) of objective value ϕ for the single-sink buy-at-bulk problem to connect X with r on the graph G_h^{up} , we can efficiently recover an integral (resp. fractional solution) of objective value at most ϕ for the problem on the original graph G .*

Likewise, we can obtain a downward directed, layered graph G_h^{down} on $(h + 1)$ -levels with edges going from top to bottom, with the same properties as above except for single-source instances instead.

3 Proof of Theorem 2 (Online Reduction to Single-Sink Instances)

There are three main steps in the proof. In Section 3.1, we describe the composite LP which is a relaxation of optimal junction-tree solutions (for technical reasons, we first need to pre-process the graph). Next, in Section 3.2, we show how to fractionally solve the LP online. Third, in Section 3.3, we show how to *partially round* the LP online. The resulting solution then decomposes as fractional solutions to different single-sink instances. Finally, we use the bounded integrality gap and the online algorithm for SS-BB to wrap up the proof in Section 3.4.

3.1 The Composite-LP Relaxation: MC-BaB LP

We first apply Theorem 7 with $h = \Theta(\log n)$ to obtain layered graphs G^{up} (resp., G^{down}) of height $O(\log n)$ where all the edges are directed upward (resp. downward); see Figure 3 for an illustration. The reason for this preprocessing is that the length of the (s_i, t_i) paths appear as a factor in our final competitive ratio and the above step bounds it to a logarithmic factor. Recall that the graph G^{up} (resp., G^{down}) approximately preserves the single-sink (resp., single-source) solutions for any set of terminals and any root. After this step, we can imagine that all the roots (of the single-sink instances we will solve) are vertices in level 0, and all the terminals will be vertices in level $h = \Theta(\log n)$. For clarity of presentation, we refer to the root and terminal vertices by the same name in both G^{up} and G^{down} (even though the graphs are completely disjoint). Overloading notation, let V denote the vertices in level 0 in both G^{up} and G^{down} , and let E be the union of the edge sets of G^{up} and G^{down} . Furthermore, the cost c_e and length ℓ_e of these edges are inherited from Theorem 7.

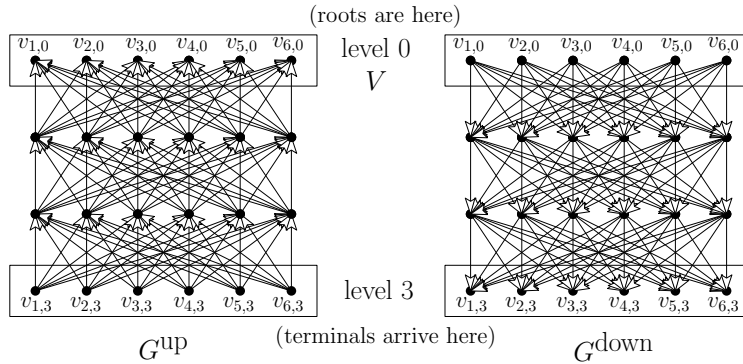


Figure 3: The graphs G^{up} and G^{down} with $h = 3$, where the original graph G has 6 vertices $\{v_1, v_2, \dots, v_6\}$.

Now, using the junction-tree decomposition with approximation factor α , we get the following lemma.

Lemma 8. *There exists a set $R^* \subseteq V$ of root vertices, a partition $\Pi^* := \{\pi_r : r \in R^*\}$ of the terminal-pairs in \mathcal{X} , a collection of in-trees $\{T_r^{\text{up}} : r \in R^*\}$ rooted at r in G^{up} , and a collection of out-trees $\{T_r^{\text{down}} : r \in R^*\}$ rooted at r in G^{down} such that*

- (i) *Each $(s_i, t_i) \in \mathcal{X}$ belongs to π_r for some $r \in R^*$.*
- (ii) *For each $r \in R^*$, the in-tree T_r^{up} is a feasible solution to the single-sink buy-at-bulk problem connecting $\{s_i : (s_i, t_i) \in \pi_r\}$ to r in G^{up} ; likewise, the out-tree T_r^{down} is a feasible solution to the single-source buy-at-bulk problem connecting r to $\{t_i : (s_i, t_i) \in \pi_r\}$ in G^{down} .*
- (iii) *The sum of objective values of the single-source and single-sink solutions is at most $O(\alpha \log n) \cdot \text{Opt}(\mathcal{I})$.*

Proof: Since the junction-tree approximation ratio of the given instance is α , there exists a junction-tree solution given by a set of roots R^* and a partition $\Pi^* := \{\pi_r : r \in R^*\}$ such that the total objective value of all the single-sink and single-source junction-trees is at most $\alpha \text{Opt}(\mathcal{I})$. Moreover, by Theorem 7, because we choose the height $h = \Theta(\log n)$, the objective value of each single-sink and single-source solution in G increases by a factor of $O(\log n)$ in G^{up} and G^{down} respectively. Thus, the overall objective value of the resulting junction-trees in the graphs G^{up} and G^{down} is $O(\alpha \log n) \cdot \text{Opt}(\mathcal{I})$. ■

The above lemma motivates the LP relaxation given in Fig. 4 which seeks to assign each (s_i, t_i) pair to some rooted instance, and then minimizes the total fractional objective value of the rooted instances. Each individual rooted instance is represented by an inner-LP (see the boxed constraints in Fig. 4).

$$\text{minimize } \sum_{r \in V} \sum_{e \in E} c_e x_e^r + \sum_{(s_i, t_i) \in \mathcal{X}} \sum_{r \in R} \sum_{e \in E} \ell_e \left(f_{(e, s_i)}^r + f_{(e, t_i)}^r \right) \quad (\text{MC-BaB LP})$$

$$\text{s.t. } \sum_{r \in V} z_{ir} \geq 1 \quad \forall i \quad (2a)$$

$$z_{ir} \geq 0 \quad (2b)$$

$$\boxed{\begin{array}{l} \{f_{(e, s_i)}^r\} \text{ define a flow from } s_i \text{ to } r \text{ of value } z_{ir} \text{ in } G^{\text{up}} \quad \forall i, \forall r \in V \end{array}} \quad (2c)$$

$$\boxed{\begin{array}{l} \{f_{(e, t_i)}^r\} \text{ define a flow from } r \text{ to } t_i \text{ of value } z_{ir} \text{ in } G^{\text{down}} \quad \forall i, \forall r \in V \end{array}} \quad (2d)$$

$$\boxed{\begin{array}{l} f_{(e, s_i)}^r \leq x_e^r \quad \forall i, \forall e, \forall r \in V \end{array}} \quad (2e)$$

$$\boxed{\begin{array}{l} f_{(e, t_i)}^r \leq x_e^r \quad \forall i, \forall e, \forall r \in V \end{array}} \quad (2f)$$

$$\boxed{\begin{array}{l} x_e^r \geq 0, f_{(\cdot)}^r \geq 0 \end{array}} \quad (2g)$$

Figure 4: Composite LP for MC-BB. Equations (2a) and (2b) form the outer-LP; (2c)-(2g) form the inner-LPs.

In the LP, z_{ir} denotes the extent to which the pair (s_i, t_i) chooses root r to route its flow. Within each inner-LP corresponding to a root r , $\{x_e^r\}$ are the variables which denotes whether edge e is used to route flow in the corresponding rooted instance, and $f_{(e, s_i)}^r$ (resp. $f_{(e, t_i)}^r$) denotes the amount of flow s_i sends (resp., t_i receives) along e to (resp., from) root r . Observe that if the z_{ir} variables are integral, then the inner-LP corresponding to every root r constitutes a feasible fractional solution to (SS-BaB LP) for the single-sink instance $\mathcal{I}' = (G^{\text{up}}, \mathcal{X}')$ where $\mathcal{X}' = \{(s_i, r) : z_{ir} = 1\}$ and the single-source instance $\mathcal{I}'' = (G^{\text{down}}, \mathcal{X}'')$ where $\mathcal{X}'' = \{(r, t_i) : z_{ir} = 1\}$. The next lemma, which bounds the optimal value of the MC-BaB LP, follows directly from Lemma 8.

Lemma 9. *The optimum value of (MC-BaB LP) is $O(\alpha \log n) \cdot \text{Opt}(\mathcal{I})$.*

3.2 An online fractional algorithm for the MC-BaB LP

Theorem 10. *There is a randomized, polynomial-time online algorithm that returns a feasible fractional solution for (MC-BaB LP) of value at most $O(\alpha \log^3 n) \cdot \text{Opt}(\mathcal{I})$.*

In the remainder of the subsection, we prove the above theorem. We remark that the overall reduction uses Theorem 10 as a black-box and the time-constrained reader can skip the proof and move to Section 3.3.

To simplify the exposition, we assume that we know the cost of an optimal solution $\text{Opt}(\mathcal{I})$ up to a constant factor, using a standard doubling trick¹¹. Once we know $\text{Opt}(\mathcal{I})$, by re-scaling all the parameters

¹¹Suppose our online algorithm has a competitive ratio of α , and the true cost of an optimal solution is c^* . Then, we begin with an initial guess for the optimal cost, and run the online algorithm assuming this guess is the correct estimate for c^* . If our online algorithm fails to find a feasible solution of cost at most α times the current guess, we double our guess and run the online algorithm again. Eventually, our guess will exceed the optimal cost c^* by at most a factor of two, and for this guess, the algorithm

in the problem, we may assume that it equals 1. Next, we delete any edge in G^{up} or G^{down} that has cost c_e or length ℓ_e larger than 1 as such edges cannot participate in any optimal solution. Subsequently, we initialize all x_e^r variables to $1/n^5$. Likewise, we initialize all z_{ir} variables to $1/n^5$ and also send an initial flow of $1/n^5$ from each s_i to r in G^{up} on an arbitrary flow path from s_i to r and likewise from r to t_i in G^{down} . This setting ensures that the cost of the initial solution is $o(1)$.

In the following, we partition the edge set E into disjoint sets $\{E_j : 0 \leq j \leq h-1\}$, where E_j denotes the set of edges in E between levels j and $j+1$. Furthermore, for clarity of exposition, we describe a ‘continuous-time’ version of the algorithm where we increase the variables as a function of time. We note that this algorithm can easily be discretized for a polynomial-time¹² implementation. The algorithm is given as Algorithm 1.

Algorithm 1 Online Fractional Algorithm for (MC-BaB LP)

When a terminal-pair (s_i, t_i) arrives, we update the LP solution using the following steps:

- (1) Let R_i denote the set of roots r in level 0 such that s_i is connected to r in G^{up} and r is connected to t_i in G^{down} . For each $r \in R_i$, initialize a flow of value $1/n^5$ using any arbitrary flow path from s_i to r in G^{up} and likewise from r to t_i in G^{down} . Also set $z_{ir} = 1/n^5$ for these roots.
 - (2) Repeat the following while $\sum_{r \in R_i} z_{ir} < 1$:
 - (a) Call an edge $e \in E$ **tight** for root r if $x_e^r = f_{(e,s_i)}^r$ or $x_e^r = f_{(e,t_i)}^r$.
 - (b) **Edge Update:** For all *tight* edges $e \in E$, **update** x_e^r at the rate $\frac{dx_e^r}{dt} := \frac{x_e^r}{c_e}$.
 - (c) **Flow Update:** Solve the following min-cost max-flow problem for each $r \in R_i$: maximize Δ such that
 - there exists a flow $\{g_{(e,s_i)}^r\}$ sending Δ units of flow from s_i to r in G^{up} ,
 - there exists a flow $\{g_{(e,t_i)}^r\}$ sending Δ units of flow from r to t_i in G^{down} ,
 - *Capacity constraints:* $g_{(e,s_i)}^r \leq \text{TheAlgorithm}(\text{Algorithm 1}).x_e^r/c_e$ and $g_{(e,t_i)}^r \leq x_e^r/c_e$ for all tight edges e ,
 - *Cost constraint:* $\sum_e \ell_e \cdot g_{(e,s_i)}^r \leq z_{ir}$ and $\sum_e \ell_e \cdot g_{(e,t_i)}^r \leq z_{ir}$.
 - (d) **Update** $f_{(e,s_i)}^r$ at the rate $\frac{df_{(e,s_i)}^r}{dt} := g_{(e,s_i)}^r$, and $f_{(e,t_i)}^r$ at the rate $\frac{df_{(e,t_i)}^r}{dt} = g_{(e,t_i)}^r$ for all e , and update z_{ir} at the rate $\frac{dz_{ir}}{dt} = \Delta$.
-

We increase the x variables on tight edges at a rate inversely proportional to their cost, similar to the well-known online set cover algorithm [3]. However, the ‘‘flow constraints’’ are not pure packing (or covering) constraints and there is no general-purpose way of handling them. Indeed, we determine the rate of increase of the flow variables by solving an auxiliary min-cost max-flow subroutine which routes incremental flows of equal value from s_i to r in G^{up} and from r to t_i in G^{down} respecting capacity constraints (i.e., for edges that are tight, the incremental flow is at most the rate of increase of x). This maintains feasibility in the inner LP. Moreover, to bound the rate of increase in objective, we enforce that the total length of the incremental flow

will compute a feasible solution of cost at most $2\alpha c^*$. Moreover, since our guesses double every time, the total cost of the edges bought by the online algorithm over all the runs across different guesses is at most $\Theta(\alpha c^*)$.

¹²The polynomial is in the size of the input to this algorithm, which for some of our algorithms/results is quasi-polynomial in the size of the actual problem instance as stated in the introduction.

is at most z_{ir} (this is the “cost” constraint in the min-cost max-flow problem). We stress that the incremental flows from the auxiliary problem dictate the *rate* at which we increase the original flow variables in the LP. The final solution is feasible since the algorithm runs until the outer-LP constraint is satisfied.

First, note that the total cost of initialization is $o(1)$ over all the edge and flow variables. So it suffices to bound the cost of the updates. The next lemma relates the total cost of the updates to the total time τ for which the algorithm runs, and the subsequent lemma bounds τ in terms of $\text{Opt}(\mathcal{I})$.

Lemma 11. *The LP objective value at the end of the above algorithm is $O(\log n) \cdot \tau$, where τ is the (continuous) time for which the algorithm runs.*

Proof: We show that at any time t , the rate of the increase of the LP objective value in the algorithm is at most $O(\log n)$; this proves the lemma.¹³ The objective increases because of increase in x variables and flow variables f ; we bound these separately.

We first upper bound the objective increase due to the changes in the x variables. Fix a level j and let E_j^{tgt} denote the set of tight edges in E_j at time t . By definition, $\sum_{e \in E_j^{\text{tgt}} \cap G^{\text{up}}} x_e^r$ (resp., $\sum_{e \in E_j^{\text{tgt}} \cap G^{\text{down}}} x_e^r$) equals the total flow on these edges for the pair (s_i, t_i) . Since the edges in E_j form a cut separating s_i from t_i , the total flow across this cut is at most z_{ir} . Since $\sum_r z_{ir} < 1$, we have $\sum_r \sum_{e \in E_j^{\text{tgt}}} x_e^r < 2$. Now, the rate of increase of each such tight edge is precisely x_e^r / c_e , which implies that the total rate of increase of the LP value due to the increase of x is at most

$$\sum_r \sum_{e \in E_j^{\text{tgt}}} c_e \frac{dx_e^r}{dt} \leq \sum_r \sum_{e \in E_j^{\text{tgt}}} x_e^r \leq 2.$$

Summing over all levels gives the desired $O(\log n)$ bound.

Next, we upper bound the objective increase due to the changes in the f variables. When these variables are updated, the total rate of increase of the objective due to the lengths of the (s_i, r) and (r, t_i) flow paths is at most z_{ir} — this is precisely the “cost” constraint in the auxiliary flow problem. Hence the total rate of increase of flow lengths is at most 2, completing the proof. ■

Given the above lemma, we are left to relate τ to $\text{Opt}(\mathcal{I})$ in order to complete the proof of Theorem 10.

Lemma 12. *The time duration τ of the above algorithm satisfies $\tau = O(\alpha \log^2 n) \cdot \text{Opt}(\mathcal{I})$.*

We will need several new definitions and auxiliary lemmas in order to prove Lemma 12. Recall from Lemma 8 that we can assume that the solution that we are comparing against is the set of junction-trees defined by T_r^{up} and T_r^{down} for $r \in R^*$. Also, recall that the terminal-pairs are partitioned by the groups $\Pi^* = \{\pi_r : r \in R^*\}$. For every $(s_i, t_i) \in \mathcal{X}$, let $P_{s_i}^*$ denote the path from s_i to the root r in T_r^{up} such that $(s_i, t_i) \in \pi_r$. Similarly, let $P_{t_i}^*$ denote the path from r to t_i in T_r^{down} . Let $\ell(P) = \sum_{e \in P} \ell_e$ for any path P . Lemma 8 asserts that

$$\sum_{r \in R^*} \left(\sum_{e \in T_r^{\text{up}} \cup T_r^{\text{down}}} c_e + \sum_{(s_i, t_i) \in \mathcal{X}} (\ell(P_{s_i}^*) + \ell(P_{t_i}^*)) \right) = O(\alpha \log n) \cdot \text{Opt}(\mathcal{I}) \quad (3)$$

To bound τ against the optimal junction-tree solution, we use two sets of *charging clocks*:

¹³We remark that the “ $\log n$ ” corresponds to the number of levels in G_h justifying the preprocessing step before the LP description.

- We maintain an *edge clock* on every (e, r) pair such that $e \in T_r^{\text{up}}$ or $e \in T_r^{\text{down}}$, i.e., if e is used by the optimal junction-tree solution in the single-source (or single-sink) instance corresponding to r . In particular, note that if an edge e is in multiple junction-trees, then it has a separate clock for each such tree.
- We maintain a *terminal clock* on every terminal-pair $(s_i, t_i) \in \mathcal{X}$.

The crucial invariant that we maintain is the following: *at any time instant t , at least one clock “ticks,” i.e., augments its counter at unit rate.* The overall goal would then be to bound the total time for which all the charging clocks can cumulatively tick.

First, we describe the rule for the ticking of the clocks. Fix a time t , and let the terminal-pair (s_i, t_i) be the pair that is active at time t . Let r denote the root vertex which (s_i, t_i) has been assigned to in the optimal junction-tree solution from Lemma 8, i.e., $(s_i, t_i) \in \pi_r$. Now, consider the flow-paths $P_{s_i}^*$ in G^{up} and $P_{t_i}^*$ in G^{down} . We can have one of two situations:

- If any variable x_e^r is tight for any edge $e \in P_{s_i}^* \cup P_{t_i}^*$ at time t , then the edge clock on the pair (e, r) ticks at time t . If there are multiple such edges, then all the corresponding clocks tick.
- Otherwise, both paths are free of tight edges. In this case, the terminal clock for (s_i, t_i) ticks at time t .

Lemma 13. *For any pair (e, r) such that $e \in T_r^{\text{up}} \cup T_r^{\text{down}}$, its edge clock ticks for $O(c_e \log n)$ time.*

Proof: Notice that x_e^r is initialized to $1/n^5$ for all roots r , and increases at the rate

$$\frac{dx_e^r}{dt} = \frac{x_e^r}{c_e} \quad (4)$$

at all times when the edge clock on (e, r) ticks. To see why, consider a time t when the clock on (e, r) ticks, and let (s_i, t_i) denote the active terminal-pair at time t . It must be that (i) (s_i, t_i) has been assigned to root r in Π^* , and (ii) either $x_e^r = f_{(e, s_i)}^r$ or $x_e^r = f_{(e, t_i)}^r$. But in this case, we increase such variables at rate x_e^r/c_e in our algorithm (Step (2a)). Therefore, we can infer that the value of x_e^r would be 1 after the edge clock on e has ticked for time $O(c_e \log n)$. But clearly, e cannot be a tight edge for any subsequent terminal-pair (s_i, t_i) once x_e reaches 1; therefore, the edge clock on (e, r) ticks for $O(c_e \log n)$ time overall. ■

Lemma 14. *For every terminal-pair (s_i, t_i) connected by the optimal junction-tree solution through the root vertex r , the total time for which the terminal clock ticks is at most $O(\log n) \cdot \max(\ell(P_{s_i}^*), \ell(P_{t_i}^*))$.*

Proof: Recall that if the terminal clock for (s_i, t_i) is ticking at time t , then it must mean that no edge is tight on either path $P_{s_i}^*$ or $P_{t_i}^*$. In this case, we show that the variable z_{ir} increases at a fast enough rate, where r is the root (s_i, t_i) is assigned to in the optimal junction-tree, i.e., $(s_i, t_i) \in \pi_r$. We show this by exhibiting a feasible solution to the auxiliary LP considered in Step (2b) of the algorithm for root r . Indeed, send the flow from s_i to r along $P_{s_i}^*$, and likewise from r to t_i along $P_{t_i}^*$. Also set the value of Δ to be $z_{ir} / \max(\ell(P_{s_i}^*), \ell(P_{t_i}^*))$. Clearly, on the edges of these flow paths, we do not have any capacity constraints since no edge is tight. So, the only constraints are the cost constraints which are satisfied by the choice of Δ . Hence, the rate of increase of z_{ir} is at least

$$\frac{dz_{ir}}{dt} \geq \frac{z_{ir}}{\max(\ell(P_{s_i}^*), \ell(P_{t_i}^*))} \quad (5)$$

at all times when the terminal clock on (s_i, t_i) ticks. This proves the claim, for otherwise the variable z_{ir} would have reached 1, and the algorithm would have completed processing (s_i, t_i) . ■

Since at least one clock ticks at all times, the total time clocked is at least τ , the duration of the algorithm. Lemma 13 and Lemma 14 imply that

$$\tau \leq O(\log n) \sum_{r \in R^*} \left(\sum_{e \in T_r^{\text{up}} \cup T_r^{\text{down}}} c_e + \sum_{(s_i, t_i) \in \mathcal{X}} (\ell(P_{s_i}^*) + \ell(P_{t_i}^*)) \right)$$

which together with (3) completes the proof of Lemma 12. Theorem 10 follows from Lemma 11 and Lemma 12.

3.3 Partial Online LP Rounding

We partially round the fractional solution returned by Theorem 10 to obtain integral values for only the outer-LP variables z_{ir} , i.e., each (s_i, t_i) pair is integrally assigned to a root. The inner-LP variables x and f continue to be fractional but represent *unit* fractional flow from s_i to r and r to t_i for the (s_i, t_i) pairs assigned to r . The partial rounding algorithm is given as Algorithm 2.

Algorithm 2 Online Partial Rounding Algorithm

- (1) **Initialization:** Each root chooses a threshold $\tau_r \in [1/2n, 1/(3 \log n)]$ uniformly at random.
 - (2) **Partial Rounding:** At each time, maintain the scaled solution $\tilde{x}_e^r = \min(1, x_e^r/\tau_r)$, $\tilde{f}_{(\cdot)}^r = \min(1, f_{(\cdot)}^r/\tau_r)$. Also set $\tilde{z}_{ir} = 1$ if $z_{ir} \geq \tau_r$.
-

Theorem 15. *The scaled solution (\tilde{x}, \tilde{f}) component-wise dominates a feasible solution to the outer-LP, and the expected objective value of the scaled solution (\tilde{x}, \tilde{f}) is at most $O(\alpha \log^5 n) \cdot \text{Opt}(\mathcal{I})$. Moreover, for each (s_i, t_i) , there exists at least one root r such that $\tilde{z}_{ir} \geq 1$ with probability at least $1 - 1/n^3$.*

Proof: Since each root r chooses its threshold τ_r independently and uniformly at random from $[1/2n, 1/\log n]$, the probability that $\tilde{z}_{ir} = 1$ is at least $z_{ir} \log n$ (since $\tilde{z}_{ir} = 1$ if and only if $\tau_r \leq z_{ir}$). Since this is independent for different roots, a standard Chernoff-Hoeffding bound application (see, e.g., [30]) shows that each (s_i, t_i) pair has $\tilde{z}_{ir} = 1$ for some root r with probability at least $1 - 1/n^3$. Moreover, the expected value of any variable x_e^r is given by

$$\mathbb{E}[\tilde{x}_e^r] \leq \int_{\tau_r=1/2n}^{\log n} \frac{x_e^r}{\tau_r} \log n d\tau_r \leq O(\log^2 n) x_e^r.$$

A similar argument shows that the expected values of scaled flow variables are also bounded by $O(\log^2 n)$ times their values in the fractional solution. This shows that the expected objective value of the (\tilde{x}, \tilde{f}) solution is at most $O(\log^2 n)$ times the value of (x, f) ; by Theorem 10, the latter is at most $O(\alpha \log^3 n) \text{Opt}(\mathcal{I})$. Combining these facts gives us the desired bound on the value of the scaled solution.

It remains to show that the scaled solution dominates a feasible solution to the LP. To this end, fix some root r and let \mathcal{X}_r denote the set of (s_i, t_i) pairs for which $z_{ir} = 1$. We need to show that installing capacities of $\{\tilde{f}_{(e, s_i)}^r\}$ on the edges can support unit flow from s_i to r in G^{up} for all $(s_i, t_i) \in \mathcal{X}_r$. Suppose

for contradiction that there is a cut Q separating s_i from r of capacity strictly smaller than 1. This implies that every edge $e \in Q$ must have $f_{(e,s_i)}^r \leq \tau_r$; otherwise, we would have an edge e with $\tilde{f}_{(e,s_i)}^r = 1$, which contradicts our assumption on the cut capacity. But then the value of the min-cut is precisely $\left(\sum_{e \in Q} f_{(e,s_i)}^r\right) / \tau_r$, which must be at least 1 because of the following two observations: (i) we know that $\{f_{(e,s_i)}^r\}$ is a feasible flow from s_i to r of value z_{ir} and hence it must be that $\sum_{e \in Q} f_{(e,s_i)}^r \geq z_{ir}$, and (ii) since $\tilde{z}_{ir} = 1$, it must be that $z_{ir} \geq \tau_r$. This contradicts the assumption that the cut capacity is strictly smaller than 1. A similar argument shows that the variables $\{\tilde{f}_{(e,t_i)}^r\}$ can support unit flow from r to t_i for every (s_i, t_i) with $\tilde{z}_{ir} = 1$. ■

3.4 Wrapping up: Invoking the Single-Sink Online Algorithm

We are now ready to put all the pieces together and present our overall online multicommodity buy-at-bulk algorithm as Algorithm 3. SingleSinkAlg is the online algorithm for SS-BB alluded to in point (iii) of the statement of Theorem 2.

Algorithm 3 Online Multicommodity Buy-at-Bulk Algorithm

when (s_i, t_i) arrives

- (1) **update** the fractional solution of the composite LP using the algorithm (Algorithm 1, Section 3.2).
 - (2) **partially round** the solution using algorithm in (Fig. 2, Section 3.3).
 - (3) **if** $(\exists r : z_{ir} \geq 1)$: send both s_i and t_i to the instance of SingleSinkAlg with root r .
 - (4) **else**: buy a trivial shortest path between s_i and t_i on the metric $(c + \ell)$ and route along this path
-

Clearly the algorithm produces a feasible solution; so we now argue about the expected objective value. Fix an (s_i, t_i) pair. Since the probability that a terminal-pair is not assigned to a root is $\leq 1/n^3$ (by Theorem 15), the expected total contribution of such unassigned terminal-pairs is $\leq 1 = \text{Opt}(\mathcal{I})$. For a root r , let π_r be the terminal-pairs assigned to r . We know that (\tilde{x}, \tilde{f}) restricted to π_r dominates a feasible solution in (SS-BaB LP). Letting LP_r denote the contribution of this restriction to the overall LP value, we get $\sum_r LP_r = O(\alpha \log^5 n) \cdot \text{Opt}(\mathcal{I})$. By the integrality gap condition, we get that Opt_r , i.e. the *integral* optimum objective value of the instance generated by r and π_r , is at most $\beta \cdot LP_r$. (Here we are using the fact from Theorem 7 that moving to the layered instance does not increase the integrality gap.) The objective value of the solution produced by SingleSinkAlg is at most $\gamma \cdot \text{Opt}_r$, where γ is the competitive ratio of SingleSinkAlg. Putting these observations together, we conclude that the overall objective value of the solution returned by the online algorithm is $O(\alpha\beta\gamma \log^5 n) \cdot \text{Opt}(\mathcal{I})$. This completes the proof of Theorem 2.

4 Online Directed Buy-at-Bulk

In this section, we prove Theorem 5. A natural approach is to use the reduction given by Theorem 2. To this end, we need to establish the following: the existence of a junction-tree scheme with a good approximation; a good upper bound on the integrality gap for single-sink instances of the LP given in Section 2; and an online algorithm for single-sink instances with a good competitive ratio.

Extending the work of Chekuri et al. [13], Antonakopoulos [5] shows the existence of a junction-tree scheme with approximation $O(\sqrt{k})$. Unfortunately, the integrality gap of the LP relaxation is not very well understood even for Steiner tree instances; [35] gives an $\Omega(\sqrt{k})$ lower bound¹⁴ on the integrality gap for the Steiner tree problem and no suitable upper bound is known. We overcome this difficulty as follows. Instead of working with general graphs, we pre-process the instance and obtain a tree-like graph for which we can show that the LP has a good integrality gap. Finally, we give the first non-trivial online algorithm for the directed single-sink buy-at-bulk problem. These results, together with our reduction (Theorem 2), imply the online algorithm for MC-D-BB.

We devote the rest of this section to the proof of Theorem 5; to aid the reader, we restate the theorem below.

Theorem 16. *For any constant $\varepsilon > 0$, there is a $O(k^{\frac{1}{2}+\varepsilon} \text{polylog}(n))$ -competitive, polynomial time randomized online algorithm for the general buy-at-bulk problem.*

Pre-processing step. We first give our reduction from general instances of the directed buy-at-bulk problem to much more structured instances; the reduction loses a factor of $O(k^{\frac{1}{2}+\varepsilon})$ in the approximation ratio.

Let $h = \lceil 1/\varepsilon \rceil$. Given an instance $\mathcal{I} = (G, \mathcal{X})$ of the directed buy-at-bulk problem, we map it to a tree-like instance $\mathcal{J} = (H, \mathcal{X})$ as follows. We start by applying Theorem 7 to G to obtain the graphs G^{up} and G^{down} ; recall that these graphs are layered $(h + 1)$ -level graphs with n vertices (corresponding to the vertices in G) in each level, and the levels are numbered $0, 1, \dots, h$ with 0 being called the root level. The graph G^{up} has edges directed from higher numbered levels to lower numbered levels, and G^{down} has edges in the opposite direction. To facilitate the construction of the graph H , we now create n trees from G^{up} and n trees from G^{down} as follows.

For every “root vertex” r at level 0 in G^{up} (resp. G^{down}), the tree T_r^{up} (resp. T_r^{down}) is constructed as follows:

- The 0th layer of T_r^{up} has just one vertex – the root r .
- For each i such that $1 \leq i \leq h$, the i -th layer of T_r^{up} contains all $(i + 1)$ -length tuples (r, v_1, \dots, v_i) where v_j is a vertex present in the j -th layer of G^{up} .
- For every edge $e = (v_i, v_{i-1}) \in G^{\text{up}}$, there is an arc from $(r, v_1, \dots, v_{i-1}, v_i)$ to (r, v_1, \dots, v_{i-1}) inheriting the same cost c_e and length ℓ_e .

Therefore each tree T_r^{up} is an in-arborescence, with all edges directed towards the root. The tree T_r^{down} is constructed analogously except all edges are directed away from the root. In the following, we use the term *leaves* to refer to the vertices on layer h of these trees.

After performing the above operation for every root vertex r in level 0 of G_r^{up} and G_r^{down} , we have $2n$ trees. Then the final graph H is obtained as follows (see Figure 5). For each root $r \in V$, we first add an arc from the root of T_r^{up} to T_r^{down} of zero cost and length. Finally, for every (s_i, t_i) pair, we add the vertices s_i and t_i to H and the following arcs connecting them to the trees: for each tree T_r^{up} , we add an arc from s_i to each leaf of T_r^{up} of the form (r, v_1, \dots, v_h) with $v_h = s_i$; for each tree T_r^{down} , we add an arc to t_i from each leaf of T_r^{down} of the form (r, v_1, \dots, v_h) with $v_h = t_i$. These new arcs have zero cost and length (i.e., $c_e = \ell_e = 0$).

This completes the construction of H . Note that the graph H has $n^{O(h)}$ vertices and a similar number of edges. Our new instance is $\mathcal{J} = (H, \mathcal{X})$ and we will apply Theorem 2 to this instance.

We first relate the objective values of \mathcal{J} and \mathcal{I} .

¹⁴However, in these instances, n is exponentially large in k . So, they do not rule out a $\text{polylog}(n)$ upper bound.

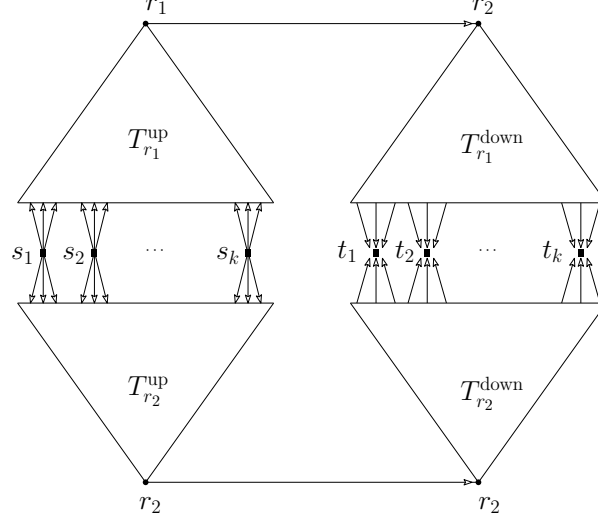


Figure 5: Construction of graph H .

Lemma 17. *Every feasible solution for \mathcal{J} is a junction-tree solution.*

Proof: Note that any (s_i, t_i) path in H has the following structure: s_i connects to a leaf node of T_r^{up} for some $r \in V$, then continues to the root r , then traverse the edge to the root of T_r^{down} , then goes down to a leaf of T_r^{down} and finally connects to t_i . Thus, for any feasible solution for \mathcal{J} , the (s_i, t_i) pairs can be partitioned based on the root r through which they connect. ■

Lemma 18. *Any feasible solution for \mathcal{J} can be mapped to a feasible solution — in fact, a junction-tree solution — for \mathcal{I} of equal or smaller objective value.*

Proof: Note that from the previous lemma, any feasible solution S in \mathcal{J} is a junction-tree solution. Therefore, there is a partition Π_S of the (s_i, t_i) pairs depending on which root vertex they are using to connect. Moreover, it follows from our construction of the trees in H that any edge in T_r^{up} (resp. T_r^{down}) corresponds to an edge in G^{up} (resp. G^{down}). Therefore, if we map each edge appearing in solution S to its corresponding edge in G^{up} or G^{down} , we obtain a mapping from each junction tree of S rooted at r to a junction tree in $G^{\text{up}} \cup G^{\text{down}}$ rooted at r that is connecting the same subset of pairs. Finally, by Theorem 7, each junction tree in $G^{\text{up}} \cup G^{\text{down}}$ rooted at r can be mapped, without increasing the objective value, to a junction tree in G rooted at r that is connecting the same subset of pairs. This completes the proof of the lemma. ■

Lemma 19. $\text{Opt}(\mathcal{J}) \leq O(hk^{1/h})\text{Opt}_{\text{junc}}(\mathcal{I})$, where $\text{Opt}_{\text{junc}}(\mathcal{I})$ is the objective value of an optimal junction-tree solution for \mathcal{I} .

Proof: Consider the optimal junction tree solution for \mathcal{I} . Let the optimum partition be $\Pi = (\pi_{r_1}, \dots, \pi_{r_q})$ where $R^* = \{r_1, r_2, \dots, r_q\}$ is the set of roots of the junction trees. For each $r \in R^*$, let $X_r = \{s_i : (s_i, t_i) \in \pi_r\}$ be the set of sources of π_r and Y_r be the corresponding sinks. From Theorem 7, we know that the optimum objective value of any one single-sink problem connecting X_r to r in G^{up} is at most $O(hk^{1/h})$ times the objective value of the optimum solution connecting each source in X_r to r . An analogous upper bound holds for every optimal single-source solution connecting r to each sink in Y_r . Therefore, we get that the total sum of objective values of each of the junction trees in G^{up} and G^{down} is at most

$O(hk^{1/h})\text{Opt}_{\text{junc}}(\mathcal{I})$. Now notice that any solution S_G for (G^{up}, X_r) can easily be “simulated” by a solution S_T in the tree T_r^{up} : indeed, for every root-vertex path $(r, v_1, v_2, \dots, v_i)$ in the solution S_G , include the edge from $(r, v_1, v_2, \dots, v_i)$ to $(r, v_1, v_2, \dots, v_{i-1})$ in S_T (recall the vertices in T_r^{up} exactly correspond to such root-vertex paths). It is easy to see that the objective value of the solution S_T in T_r^{up} is the same as that of S_G . Similarly, any solution for (G^{down}, Y_r) can be simulated in T_r^{down} with the same objective value. It follows that there is a feasible solution in \mathcal{J} of objective value at most $O(hk^{1/h})\text{Opt}_{\text{junc}}(\mathcal{I})$. ■

Corollary 20. $\text{Opt}(\mathcal{J}) \leq O(k^{\frac{1}{2}+\varepsilon})\text{Opt}(\mathcal{I})$.

Proof Sketch: Antonakopoulos [5] shows that there exists a junction-tree solution of cost at most $O(\sqrt{k})\text{Opt}(\mathcal{I})$. The corollary follows from this work and the fact that we set $h = \Theta(1/\varepsilon)$. ■

Now we are ready to show that the new instance \mathcal{J} has the properties required by the reduction, i.e., Theorem 2 can be applied. In the following lemma, a single-source (resp. single-sink) *sub-instance* $\mathcal{J}' = (H, \mathcal{T}, v)$ of $\mathcal{J} = (H, \mathcal{X})$ is a single-source (resp. single-sink) instance of the following form: the graph is the same as in \mathcal{J} , namely H ; the set of terminals \mathcal{T} is a subset of the sources (resp. sinks) of \mathcal{X} ; the terminals \mathcal{T} need to be connected to a root vertex $v \in V(H) \setminus \{s_i, t_i : i \in [k]\}$.

Lemma 21. *Let \mathcal{J} be the instance described above. Let α, β, γ be as in the statement of Theorem 2. We have*

- (i) *The junction-tree approximation factor of \mathcal{J} is 1 (i.e., $\alpha = 1$).*
- (ii) *The integrality gap of (SS-BaB LP) for any single-sink/source sub-instance $\mathcal{J}' = (H, \mathcal{T}, v)$ is $O(h \log n \log k)$ (i.e., $\beta = O(h \log n \log k)$).*
- (iii) *There is a $O(h^2 \log^2 n \log k)$ -competitive algorithm for any single-sink/source sub-instance $\mathcal{J}' = (H, \mathcal{T}, v)$ (i.e., $\gamma = O(h^2 \log^2 n \log k)$).*

Proof: Property (i) follows from Lemma 17. Thus we focus on proving (ii) and (iii). In the following, we assume that we are working with a single-sink sub-instance \mathcal{J}' of \mathcal{J} ; the proof is very similar for single-source sub-instances and we omit it.

In order to show (ii) and (iii), we will map the sub-instance $\mathcal{J}' = (H, \mathcal{T}, v)$ to an instance of the *group Steiner tree* problem on a tree as follows. Since $v \in V(H) \setminus \{s_i, t_i : i \in [k]\}$, we have $v \in T_r^{\text{up}} \cup T_r^{\text{down}}$ for some r . We first consider the case when $v \in T_r^{\text{up}}$. In order to define the tree of the group Steiner tree instance, we start with the subtree T_v of T_r^{up} rooted at v . We add the following ‘dangling’ edges to T_v for each source $s_i \in \mathcal{T}$: for each leaf vertex $u \in T_r^{\text{up}}$ such that s_i has an edge in T_r^{up} to u , we add a new vertex u' to T_v and connect it to u . Let T be the resulting tree. We assign weights to the edges of T as follows. Each of the old edges $e \in T_v$ receives a weight equal to c_e . Each of the new edges $uu' \in E(T) \setminus E(T_v)$ receives a weight equal to $\ell(P_u)$, where P_u is the path of T_r^{up} from u to v . Finally, we define the following groups: for each source $s_i \in \mathcal{T}$, we introduce a group S_i consisting of all the new vertices $u' \in V(T) \setminus V(T_v)$ such that s_i is connected to its partner u in T_r^{up} (that is, T_r^{up} has an edge from s_i to u). In the resulting group Steiner tree instance, the goal is to connect all of the groups $\mathcal{S} = \{S_i : s_i \in \mathcal{T}\}$ to the root v using a minimum weight subtree of T ; we let (T, \mathcal{S}, v) denote this instance.

Now the key claim is that the feasible solutions to the single-sink buy-at-bulk (H, \mathcal{T}, v) are in a one-to-one correspondence with feasible solution to the group Steiner tree instance (T, \mathcal{S}, v) ; moreover, the objective value of a solution to the former is equal to the weight of the solution to the latter. To see this, consider a feasible solution \mathcal{P} for the single-sink buy-at-bulk instance. Note that, for each source $s_i \in \mathcal{T}$, \mathcal{P} has a path connecting s_i to v ; it follows from our construction of H that this path consists of an edge from

s_i to a leaf u of T_r^{up} followed by the unique path in T_r^{up} from u to v . Thus we can construct a feasible group Steiner tree solution by connecting each group S_i using the path of T from u' to v , where u' is the partner of the leaf u of T_r^{up} through which s_i connects to the root in the buy-at-bulk solution. The weight of the edge $u'u$ captures the ℓ -cost of s_i 's path and the weight of the path from u to v captures the c -cost of s_i 's path.

Moreover, we can apply the same argument to fractional solutions to the two problems and show that there is a bijection between feasible fractional solutions to (SS-BaB LP) and feasible fractional solutions to the LP relaxation for group Steiner tree of Garg, Konjevod, and Ravi [19]; as before, these corresponding solutions have the same objective values. Now the desired upper bound on the integrality gap of (SS-BaB LP) follows from the work of [19] who showed that the integrality gap of the group Steiner tree LP is $O(\log N \log K)$ where $N = \max_i |S_i|$ is the maximum size of a group and K is the number of groups. In our setting, $K \leq |\mathcal{X}| \leq k$ and $N \leq n^h$. Therefore the integrality gap is $O(h \log n \log k)$, which establishes property (ii).

Moreover, notice that the above reduction can also be used to obtain an online algorithm for the single-sink (and single-source) sub-instances. Indeed, we simply use the online group Steiner tree algorithm of Alon *et al.* [1] which has a competitive ratio $O(\log^2 N \log K) = O(h^2 \log^2 n \log k)$. This proves property (iii). ■

Now we are ready to complete the proof of Theorem 16.

Proof of Theorem 16: Given the instance $\mathcal{I} = (G, \mathcal{X})$, we construct the graph H as described above; the time taken to do so is $n^{O(h)}$. We pass the instance (H, \mathcal{X}) to Theorem 2 and, using Lemma 21, we obtain an online algorithm that, for any collection of pairs \mathcal{X} and any adversarial ordering of \mathcal{X} , returns a solution of cost $O(\text{polylog}(n)) \cdot \text{Opt}(\mathcal{J})$. By Lemma 19 and Corollary 20, we can map solutions for (H, \mathcal{X}) to solutions for (G, \mathcal{X}) . ■

We note that the approach described above also gives us new online algorithms for the *single-sink* buy-at-bulk problem on directed graphs. For single-sink instances, the junction-tree approximation is equal to 1 and thus we save a factor of \sqrt{k} .

Corollary 22. *There is a polynomial time $O(k^\varepsilon \cdot \text{polylog}(n))$ -competitive online algorithm for the single-sink (or single-source) buy-at-bulk problem on directed graphs. The competitive ratio can be improved to $\text{polylog}(n)$ if the running time can be quasi-polynomial in n .*

5 Online Single-sink, Undirected, Node-weighted Buy-at-Bulk

In this section, we prove Theorem 4. Again, we use our main theorem (Theorem 2) and reduce the multi-commodity buy-at-bulk problem to the single-sink version of the problem. We combine the reduction theorem with the following results from previous work. Chekuri *et al.* [13] show the existence of a junction-tree scheme with approximation factor $O(\log k)$. Moreover, the natural LP relaxation for the single-sink buy-at-bulk problem on graphs with node costs is also $O(\log k)$ [15]. For the single-source online algorithm, we resort to the algorithms from the previous section for the more general directed single-sink buy-at-bulk problem (see Corollary 22). We now obtain the desired result by the following parameter settings:

$$\begin{aligned} \alpha &= O(\log k) \\ \beta &= O(\log k) \\ \gamma &= O(\text{polylog}(n)) \\ T &= n^{O(\log n)}. \end{aligned}$$

6 Online Prize-Collecting Buy-at-Bulk

In the prize-collecting version of the buy-at-bulk problem, each terminal pair (s_i, t_i) also comes with a *penalty* q_i and the algorithm may choose not to serve this request and incur the penalty in the total cost. We show that our online reduction framework (Theorem 2) can be easily modified to handle prize-collecting versions as follows.

Theorem 23. *Let \mathcal{I} be a buy-at-bulk instance and suppose the three conditions of Theorem 2 hold. Then there is an $O(\alpha\beta\gamma \cdot \text{polylog}(n))$ -competitive online algorithm for the online, prize-collecting buy-at-bulk problem on \mathcal{I} with arbitrary penalties.*

Proof: We closely follow the proof of Theorem 2. The first difference is in the LP-formulation. Now, for each (s_i, t_i) pair we have an extra variable $z_{i,0}$ which indicates whether we choose to discard this pair (and pay the corresponding penalty) or not. We point out the differences with (MC-BaB LP). The new objective function is

$$\text{minimize} \quad \sum_{r \in V} \sum_{e \in E} c_e x_e^r + \sum_{(s_i, t_i) \in \mathcal{X}} \sum_{r \in R} \sum_{e \in E} \ell_e \left(f_{(e, s_i)}^r + f_{(e, t_i)}^r \right) + \sum_{(s_i, t_i) \in \mathcal{X}} q_i z_{i,0}$$

and (2a) is replaced by

$$\sum_{r \in V} z_{ir} + z_{i,0} \geq 1 \quad \forall i$$

Observe that the optimum value of this modified LP is at most $O(\alpha \log n)$ times the optimum: set $z_{i,0} = 1$ for the pairs the integral optimum solution does not connect, and for the rest apply Lemma 9. Also observe that the modified LP can be thought of the old LP on a modified instance where the graphs G^{up} and G^{down} (obtained from Theorem 7) have another vertex “0” at the root level, and each s_i has a direct path from s_i to “0” in G^{up} with total length $q_i/2$ and no fixed cost, and similarly, each t_i has a path from “0” to t_i in G^{down} with total length $q_i/2$ and no fixed cost. The rest of the proof now follows exactly as in Section 3, by also including the special vertex as a possible root while rounding to make the outer LP variables integral. ■

7 Conclusion

In this paper, we gave the first polylogarithmic-competitive online algorithms for the non-uniform multi-commodity buy-at-bulk problem. Our result is a corollary of a generic online reduction technique that we proposed in this paper for converting a multicommodity instance into several single-sink instances, which are often easier to design algorithms for. We believe that this reduction will have other applications beyond the buy-at-bulk framework, and illustrate this by showing that recent results on online node-weighted Steiner forest and online generalized connectivity directly follow from our reduction theorem. Our work also opens up new directions for future research. For instance, our algorithm for the node-weighted problem runs in quasi-polynomial time, and a concrete open question is to get a polynomial-time polylogarithmic-competitive algorithm for the SS-N-BB problem (this suffices for MC-N-BB as well by our main theorem). Another technical question concerns non-uniform demands. While our algorithm can be extended to the case of non-uniform demands, the approximation ratio incurs an additional $O(\log D)$ factor, where D is the ratio of the largest to the smallest demand. It would be interesting to eliminate this dependence on D since the corresponding offline results do not have this dependence. More generally, a broader question is to investigate other mixed packing-covering LPs that can be solved and rounded online.

Acknowledgements

D. Panigrahi is supported in part by NSF Award CCF-1527084, a Google Faculty Research Award, and a Yahoo FREP Award.

References

- [1] N. Alon, B. Awerbuch, Y. Azar, N. Buchbinder, and J. Naor. A general approach to online network optimization problems. In *Proceedings, ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 577–586, 2004.
- [2] N. Alon, B. Awerbuch, Y. Azar, N. Buchbinder, and J. Naor. A general approach to online network optimization problems. *ACM Trans. on Alg.*, 2(4):640–660, 2006.
- [3] N. Alon, B. Awerbuch, Y. Azar, N. Buchbinder, and J. Naor. The online set cover problem. *SIAM J. Comput.*, 39(2):361–370, 2009.
- [4] M. Andrews. Hardness of buy-at-bulk network design. In *Proceedings, IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 115–124, 2004.
- [5] S. Antonakopoulos. Approximating directed buy-at-bulk network design. In *Proceedings, Workshop on Approximation and Online Algorithms (WAOA)*, pages 13–24, 2010.
- [6] B. Awerbuch and Y. Azar. Buy-at-bulk network design. In *Proceedings, IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 542–547, 1997.
- [7] Y. Azar, U. Bhaskar, L. Fleischer, and D. Panigrahi. Online mixed packing and covering. In *Proceedings, ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 85–100, 2013.
- [8] P. Berman, A. Bhattacharyya, K. Makarychev, S. Raskhodnikova, and G. Yaroslavlsev. Approximation algorithms for spanner problems and directed steiner forest. *Inform. and Comput.*, 222:93–107, 2013.
- [9] P. Berman and C. Coulston. On-line algorithms for steiner tree problems (extended abstract). In *Proceedings, ACM Symp. on Theory of Computing (STOC)*, pages 344–353, 1997.
- [10] N. Buchbinder and J. Naor. Online primal-dual algorithms for covering and packing. *Math. Oper. Res.*, 34(2):270–286, 2009.
- [11] M. Charikar, C. Chekuri, T. Cheung, Z. Dai, A. Goel, S. Guha, and M. Li. Approximation algorithms for directed steiner problems. *J. Algorithms*, 33(1):73–91, 1999.
- [12] M. Charikar and A. Karagiozova. On non-uniform multicommodity buy-at-bulk network design. In *Proceedings, ACM Symp. on Theory of Computing (STOC)*, pages 176–182, 2005.
- [13] C. Chekuri, G. Even, A. Gupta, and D. Segev. Set connectivity problems in undirected graphs and the directed steiner network problem. *ACM Transactions on Algorithms*, 7(2):18, 2011.
- [14] C. Chekuri, M. T. Hajiaghayi, G. Kortsarz, and M. R. Salavatipour. Approximation algorithms for node-weighted buy-at-bulk network design. In *Proceedings, ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1265–1274, 2007.

- [15] C. Chekuri, M. T. Hajiaghayi, G. Kortsarz, and M. R. Salavatipour. Approximation algorithms for nonuniform buy-at-bulk network design. *SIAM J. Comput.*, 39(5):1772–1798, 2010.
- [16] C. Chekuri, S. Khanna, and J. Naor. A deterministic algorithm for the cost-distance problem. In *Proceedings, ACM-SIAM Symposium on Discrete Algorithms (SODA)*, volume 7, pages 232–233, 2001.
- [17] Y. Dodis and S. Khanna. Design networks with bounded pairwise distance. *Proceedings, ACM Symp. on Theory of Computing (STOC)*, pages 750 – 759, 1999.
- [18] M. Feldman, G. Kortsarz, and Z. Nutov. Improved approximation algorithms for directed steiner forest. *J. Comput. System Sci.*, 78(1):279–292, 2012.
- [19] N. Garg, G. Konjevod, and R. Ravi. A polylogarithmic approximation algorithm for the group steiner tree problem. *J. Algorithms*, 37(1):66–84, 2000.
- [20] S. Guha, A. Meyerson, and K. Munagala. A constant factor approximation for the single sink edge installation problem. *SIAM J. Comput.*, 38(6):2426–2442, 2009.
- [21] A. Gupta, A. Kumar, M. Pál, and T. Roughgarden. Approximation via cost-sharing: A simple approximation algorithm for the multicommodity rent-or-buy problem. In *Proceedings, IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 606–615, 2003.
- [22] A. Gupta, A. Kumar, and T. Roughgarden. Simpler and better approximation algorithms for network design. In *Proceedings, ACM Symp. on Theory of Computing (STOC)*, pages 365–372, 2003.
- [23] M. Hajiaghayi, V. Liaghat, and D. Panigrahi. Near-optimal online algorithms for prize-collecting steiner problems. In *Proceedings, International Colloquium on Automata, Languages and Processing (ICALP)*, pages 576–587, 2014.
- [24] M. T. Hajiaghayi, V. Liaghat, and D. Panigrahi. Online node-weighted steiner forest and extensions via disk paintings. In *Proceedings, IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 558–567, 2013.
- [25] C. S. Helvig, G. Robins, and A. Zelikovsky. An improved approximation scheme for the group steiner problem. *Networks*, 37(1):8–20, 2001.
- [26] M. Imase and B. M. Waxman. Dynamic steiner tree problem. *SIAM J. Discrete Math.*, 4(3):369–384, 1991.
- [27] S. Korman. On the use of randomization in the online set cover problem. *M.S. thesis, Weizmann Institute of Science*, 2005.
- [28] A. Meyerson. Online algorithms for network design. In *Proceedings, ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 275–280, 2004.
- [29] A. Meyerson, K. Munagala, and S. A. Plotkin. Cost-distance: Two metric network design. *SIAM J. Comput.*, 38(4):1648–1659, 2008.
- [30] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, 1997.
- [31] J. Naor, D. Panigrahi, and M. Singh. Online node-weighted steiner tree and related problems. In *Proceedings, IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 210–219, 2011.

- [32] F. S. Salman, J. Cheriyan, R. Ravi, and S. Subramanian. Approximating the single-sink link-installation problem in network design. *SIAM J. Optimization*, 11(3):595–610, 2001.
- [33] K. Talwar. The single-sink buy-at-bulk LP has constant integrality gap. In *Proceedings, MPS Conference on Integer Programming and Combinatorial Optimization (IPCO)*, pages 475–486, 2002.
- [34] A. Zelikovsky. A series of approximation algorithms for the acyclic directed steiner tree problem. *Algorithmica*, 18(1):99–110, 1997.
- [35] L. Zosin and S. Khuller. On directed steiner trees. In *Proceedings, ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 59–63, 2002.

A Reduction to Layered Instances (Proof of Theorem 7)

In this section, we prove Theorem 7, which is an extension of Zelikovsky’s ‘height reduction lemma’ for the buy-at-bulk problem; Zelikovsky’s original Lemma was for a single metric, whereas in our setting there is both a cost and a length metric.

We prove the up-ward case; the down-ward case follows analogously. In order to simplify the notation, we remove the superscript ^{up}. For this reduction, we will adapt the notion of layered expansion of a graph, which has been in the folklore for many years and has been used recently by several papers (see, e.g., [13, 31]). The h -level *layered expansion* of G is a layered DAG G_h of $h+1$ levels (we index the level $0, 1, \dots, h$) defined as follows:

- (i) For each i such that $0 \leq i \leq h$, the vertices in level i are copies of the vertices of G ; we let v_i to denote the copy of vertex $v \in V$ at level i .
- (ii) For each i such that $1 \leq i \leq h$, there is a directed edge from every vertex in level i to every vertex in level $i-1$. The fixed cost of an edge (u_i, v_{i-1}) is given by that of the shortest directed path P_{uv}^i from u_i to v_{i-1} in G according to the metric $c_e + k^{1-i/h} \ell_e$. The length of this edge is set to be the length of the path P_{uv}^i in the ℓ metric.

We now relate the optimal objective values for the two instances. One of the directions of the reduction is straightforward.

Lemma 24. *For any root r and any set of terminals X , if there is a feasible integral/fractional solution of objective/LP value ϕ for the single-sink buy-at-bulk problem connecting X to r on the h -level layered expansion G_h , then there is a feasible integral/fractional solution of objective/LP at most ϕ for the same problem in G .*

Proof: Note that for every edge in G_h , there is a corresponding path in G with the property that the sum of edge costs and lengths on the path is at most the cost and length of the edge in G . Therefore, replacing the edges in the solution for the layered graph by the corresponding paths in G yields a feasible solution in G without increasing the overall cost and length. Notice that the same “embedding” of edges in G_h to paths in G can be applied to the fractional solution on G_h as well. This shows property (ii) of the theorem statement. ■

The more interesting direction is to show that the optimal objective value on the layered graph G_h can be bounded in terms of the optimal objective value on the original graph G . To show this, we will re-purpose the so-called “height reduction” lemma of Helvig, Robins, and Zelikovsky [25]. We restate the lemma in a form that will be useful for us.

Lemma 25. *For any in-tree T defined on the edges of G that is rooted at r and contains all the terminals in X , and for any integer $h \geq 1$, there is an in-tree T' (on the same vertices as G but over a different edge set) that is also rooted at r and contains all the terminals, and has the following properties:*

- (i) T' contains $h + 1$ levels of vertices, i.e., has height h .
- (ii) T' is an $k^{1/h}$ -ary tree, i.e., each non-leaf vertex has $k^{1/h}$ children.
- (iii) Each edge $e' = (u', v')$ in tree T' corresponds to the unique directed path $p_{e'}$ in T from u' to v' . Moreover, the number of terminals in the subtree of T rooted at u' is exactly $k^{1-i/h}$, where e' is an edge between levels $i - 1$ and i of T .
- (iv) Each edge in T is in at most $2hk^{1/h}$ such paths $p_{e'}$ for edges $e' \in T'$.

For an edge $e' \in T'$, suppose we define its cost to be the cost of the path $p_{e'}$, and its length to be the length of the path $p_{e'}$. Then it is easy to see that the overall cost $\phi_{T'}$ of tree T' is $O(hk^{1/h})$ times that of tree T ; this is due to the following implications of the above lemma: (a) the total (buying) cost of all edges in T' is at most $2hk^{1/h}$ times that of T since each edge is reused at most $2hk^{1/h}$ times, and (b) for any terminal $x \in X$, the edges on its path to the root in T' correspond to disjoint sub-paths in the unique path between x and r in T , and hence the total length cost in T' is at most that in T .

Using this lemma, we can now complete the reduction by “embedding” the tree T' in the layered graph G_h .

Lemma 26. *If there is a feasible solution of overall cost ϕ for the single-sink buy-at-bulk problem on G , then there is a feasible solution of overall cost $O(hk^{1/h}\phi)$ for the same problem on the h -level layered extension G_h .*

Proof: Let T be the union of the paths in the optimum solution on the graph G . It's easy to see that T is a directed in-tree. First, we use Lemma 25 to transform T to tree T' of height h . As noted earlier, the overall cost $\phi_{T'}$ of T' is $O(hk^{1/h}\phi)$. Now, we construct a feasible tree T_h in G_h using this solution T' as follows: consider each edge (u, v) in T' where u is at level i and v is at level $(i - 1)$. Then, include the edge (u_i, v_{i-1}) in T_h . Clearly, since T' connects all the terminals to the root, so does T_h . Moreover, notice that there is a 1-to-1 mapping between edges in T' and edges in T_h .

To bound the objective value of the subtree, we relate the objective value for each edge of the subtree in G_h to its corresponding mapped edge in T' . First, note that the overall contribution of an edge $e' = (u', v')$ between layers i and $i + 1$ towards $\phi_{T'}$ is equal to the sum of costs and $k^{1-i/h}$ times the lengths of the edges on the associated path $p_{e'}$ from u' to v' . This is because, by property (iii) of Lemma 25, the number of demands in the subtree rooted at u' is exactly $k^{1-i/h}$ and all of them traverse this edge to reach r . Next, we note that, by definition, the cost of the edge (u'_i, v'_{i+1}) between layers i and $i + 1$ in T_h is equal to the shortest directed path from u' to v' in G according to the metric $c_e + k^{1-i/h}\ell_e$. Since we chose the shortest path, we get that the buying cost of edge $(u'_i, v'_{i+1}) \in T_h$ is at most the contribution of (u', v') towards $\phi_{T'}$. Moreover, the total length cost is at most $k^{1-i/h}$ times the length of the shortest path, which is at most the fixed cost of $(u'_i, v'_{i+1}) \in T_h$ (again, this uses the fact that there are exactly $k^{1-i/h}$ terminals which route through this edge in T_h also). It therefore follows that the overall cost of the solution that we have inductively constructed in G_h is at most twice the overall cost of T' , which is at most $O(hk^{1/h})\phi$. ■

Theorem 7 follows from Lemmas 24 and 26.